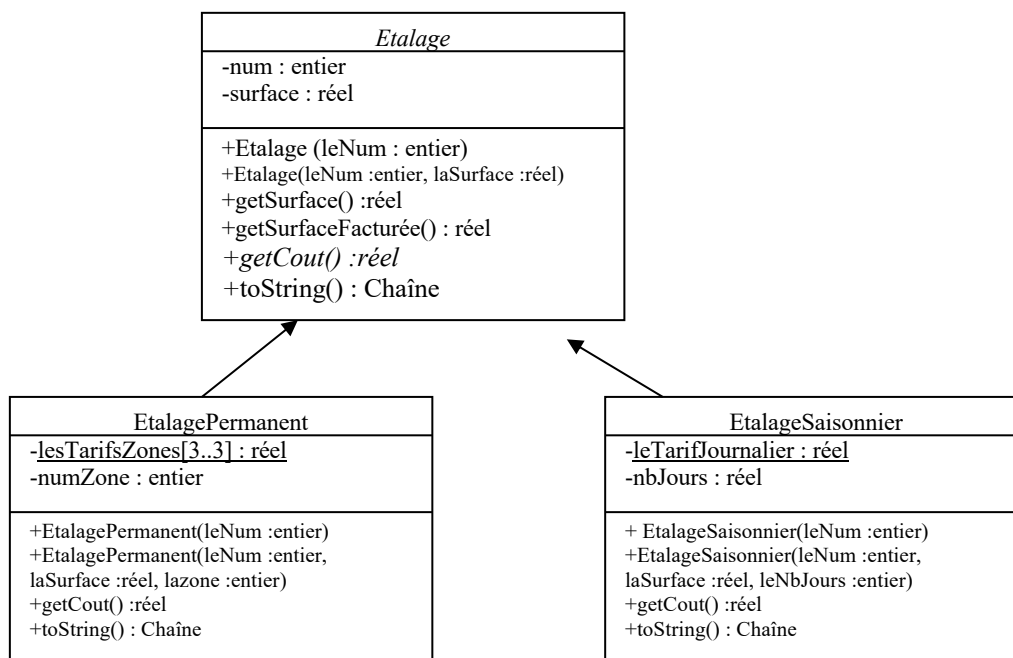


TP5 : La facturation des étalages

L'objectif de ce TP est de reprendre tous les concepts étudiés jusqu'à présent (IHM, Persistance, gestion des exceptions) sur une application déjà connue: la facturation des étalages (rappelée en annexe 1). Le diagramme de classes sur lequel va être basé le nouveau développement est présenté ci-après.



Remarque : l'italique signale que le composant (méthode ou classe) est abstrait.

TRAVAIL A FAIRE																							
1.	<p>Procéder sous NetBeans aux manipulations suivantes :</p> <p>a) Créer – le projet Java « ProjEtalage »</p> <p>b) Créer le package « etal »</p> <p>c) Créer – dans le package « etal » - les classes « Etalage », « EtalagePermanent », « EtalageSaisonnier » et « ErreurSurEtalage » partiellement définies en annexe 2</p> <p>e) Valider les classes précédentes avec la classe de test « TestEtalage ». Cette classe de test devra créer les étalages fournis dans le tableau ci-dessous et afficher, si possible, pour chaque étalage ses différentes informations ainsi que le calcul de son coût. Dans le cas où l'étalage ne peut pas être créé, l'exécution du programme ne devra pas s'interrompre mais afficher un message d'erreur.</p> <p>Liste des étalages saisonniers à créer :</p> <table> <tr> <th>Numéro</th><th>Surface</th><th>Nombre de jours</th></tr> <tr> <td>12</td><td>0,2</td><td>5</td></tr> <tr> <td>-24</td><td>0,2</td><td>5</td></tr> <tr> <td>-24</td><td>-0,2</td><td>120</td></tr> <tr> <td>12</td><td>0,2</td><td>120</td></tr> </table> <p>Liste des étalages permanents à créer :</p> <table> <tr> <th>Numéro</th><th>Surface</th><th>Numéro de la zone</th></tr> <tr> <td>24</td><td>3</td><td>1</td></tr> </table>		Numéro	Surface	Nombre de jours	12	0,2	5	-24	0,2	5	-24	-0,2	120	12	0,2	120	Numéro	Surface	Numéro de la zone	24	3	1
Numéro	Surface	Nombre de jours																					
12	0,2	5																					
-24	0,2	5																					
-24	-0,2	120																					
12	0,2	120																					
Numéro	Surface	Numéro de la zone																					
24	3	1																					

Pour offrir une interface conviviale de saisie d'un étalage, on se propose de développer les classes « SaisieEtalage » et "TestEtalage" présentées en annexe 4.

TRAVAIL A FAIRE	
2.	a) Créer – dans le package « etal » - la classe SaisieEtalage qui est une classe dérivée de la classe JPanel. Vous pouvez créer cette classe graphiquement ou à l'aide du code fourni en annexe 4. b) Créer la classe Test

On choisit de gérer la persistance des propriétés des étalages au moyen du SGBD Relationnel MySQL. Le schéma relationnel destiné à accueillir les informations est composé des deux tables suivantes :

ETALP(NumP, SURFP, NUMZONE)

ETALS(NumS, SURFS, NBJOURS)

Code champ	Type champ	Définition
NUMP	INT	Numéro identifiant un étalage permanent
SURFP	FLOAT	Surface de l'étalage permanent
NUMZONE	INT	Numéro de la zone (1, 2 ou 3) à laquelle appartient l'étalage permanent
NUMS	INT	Numéro identifiant un étalage saisonnier
SURFS	FLOAT	Surface d'un étalage saisonnier
NBJOURS	INT	Nombre de jours associés à l'étalage saisonnier

TRAVAIL A FAIRE																							
3.	Procéder sous MySQL aux manipulations suivantes :																						
	a) Créer la base « ETAL » avec les tables ci-dessus																						
	b) Créer l'utilisateur « agent » mot de passe (« agent ») avec les droits « SELECT, UPDATE et INSERT » sur la base « ETAL »																						
	d) Insérer dans les tables « ETALS » et « ETALP » les tuples décrivant les 3 étalages suivants :																						
	<table><tr><th>Type</th><th>Numéro</th><th>Surface</th><th>Nombre de jours</th><th>Zone</th></tr><tr><td>Permanent</td><td>26</td><td>4</td><td></td><td>1</td></tr><tr><td>Saisonnier</td><td>32</td><td>3</td><td>25</td><td></td></tr><tr><td>Permanent</td><td>41</td><td>0.2</td><td></td><td>2</td></tr></table>				Type	Numéro	Surface	Nombre de jours	Zone	Permanent	26	4		1	Saisonnier	32	3	25		Permanent	41	0.2	
Type	Numéro	Surface	Nombre de jours	Zone																			
Permanent	26	4		1																			
Saisonnier	32	3	25																				
Permanent	41	0.2		2																			
4.	Procéder sous NetBeans aux manipulations suivantes :																						
	a) Créer la classe « GestionConnexion » destinée à renvoyer une connexion sur la base « ETAL »																						
	b) Enrichir les classes « Etalagexxx » de méthodes destinées à gérer la persistance en respectant les spécifications fournies en annexe 5.																						
	c) Tester l'enrichissement des classes par la classe de test proposée à l'annexe 6																						
5.	Créer l'IHM destinée à afficher le coût d'un étalage et dont la maquette est présentée en annexe 7																						
6.	a) Développer dans les classes appropriées la fonctionnalité permettant d'ajouter un étalage dans la base de données.																						
	b) Modifier la classe TestEtalage.java afin qu'un clic sur le bouton "OK" permette la sauvegarde de l'étalage saisi dans la base de données "ETAL".																						

Annexe 1 : rappel du principe de facturation des étalages

Dans la ville de P. l'installation d'un étalage est soumise à une perception de droits dont le principe de calcul est donné ci-après :

- Toute occupation inférieure à 0,50 m² fera l'objet d'un droit de perception de 0,50 m²
- Pour les occupations fréquentes et répétées le tarif annuel au m² dépend de la zone de rattachement du commerce possédant l'étalage. Par exemple pour l'année 2022, le tarif pour la zone 1 est de 57,33 euros par m². Les zones (au nombre de 3 numérotées 1, 2 et 3) sont issues d'un découpage de la ville et permettent d'appliquer des tarifs dépendant de la localisation de l'établissement. Ainsi, le coût d'un étalage pour un commerce éloigné du centre ville sera très inférieur à celui d'un étalage situé dans un quartier piétonnier.
- Pour les occupations exceptionnelles le tarif au m² est journalier et est indépendant de toute zone.

Annexe 2 : la spécification des classes « Etalage », « EtalagePermanent », « EtalageSaisonnier » et « ErreurSurEtalage »

Avant-propos : les contrôles figurant dans les constructeurs des classes « Etalage », « EtalagePermanent » et « EtalageSaisonnier » doivent, en cas de détection d'erreurs, instancier un objet de la classe « ErreurSurEtalage ». Le message communiqué à l'objet représentant l'erreur sera formé du nom de la propriété suivi de la valeur fournie pour cette propriété. Par exemple si l'on tente d'instancier un étalage permanent avec les valeurs suivantes :

num : -25
surface : 64
numZone = 3

Un objet de la classe « ErreurSurEtalage » devra être créé avec comme message la chaîne de caractères :
« Numéro : -25 Surface : 64 »

```
public abstract class Etalage {
    private int num;
    private float surface;
    public Etalage (int leNum, float laSurface) throws ErreurSurEtalage
    {
        //constructeur avec contrôle sur le numéro (>0) et la surface (>0 et <50)
    }
    public Etalage (int leNum) throws ErreurSurEtalage
    {
        //constructeur avec contrôle sur le numéro (>0)
    }
    public int getNum(){
    }
    public float getSurface(){
    }
    public float getSurfaceFacturée(){
    }
    public abstract float get Cout();
    public String toString()
    {
    }
}
```

```

public class EtalagePermanent extends Etalage
{
    private static float[] lesTarifsZones={57.33f,65.25f,75.30f};
    private int numZone;
    public EtalagePermanent(int leNum) throws ErreurSurEtalage
    {
    }
    public EtalagePermanent (int leNum, float laSurface, int laZone) throws ErreurSurEtalage
    {
    }
    public float get Cout()
    {
    }
    public String toString()
    {
    }
}

public class EtalageSaisonnier extends Etalage
{
    private static float leTarifJournalier=4.23f;
    private int nbJours;
    public EtalageSaisonnier(int leNum, float laSurface, int leNbJours) throws ErreurSurEtalage
    {
    }
    public EtalageSaisonnier (int leNum) throws ErreurSurEtalage
    {
    }
    public float get Cout()
    {
    }
    public String toString()
    {
    }
}

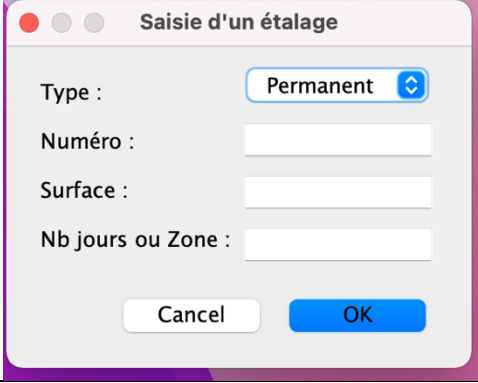
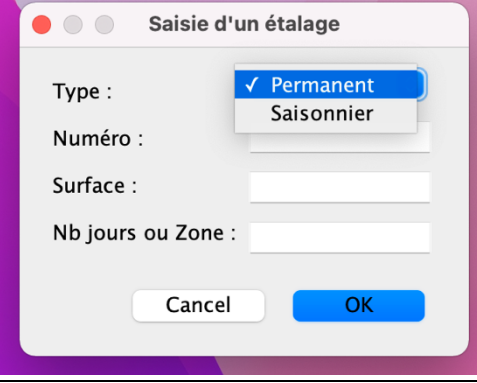
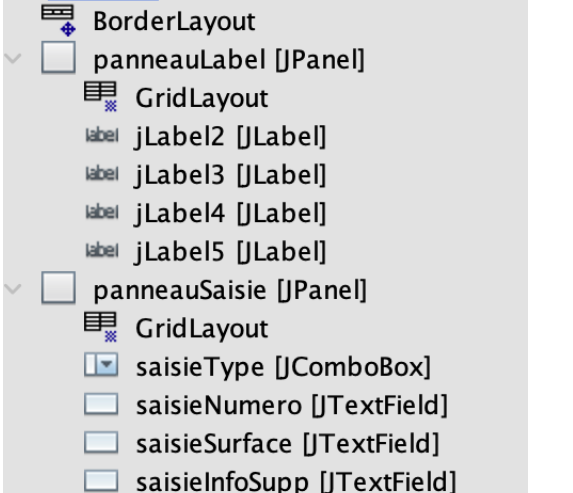
class ErreurSurEtalage extends Exception
{
    private String message;
    public ErreurSurEtalage(String leMessage)
    {
    }
    public String getMessage()
    {
        return message;
    }
    public String toString()
    {
        return "erreur d'instanciation" + message;
    }
}

```

Tableau qui contient les tarifs des 3 zones
lesTarifsZones[0] fournit le tarif de la zone n° 1, ..

Annexe 4 : La classe « SaisieEtalage » et « TestSaisieEtalage »

Conception avec l'IDE NetBeans

	
	<p>Aide à la conception :</p> <ul style="list-style-type: none"> -le panneauLabel est placé à l'Ouest du BorderLayout du panneau principal - le panneauSaisie à l'Est <p>Un click sur le bouton OK, permettra :</p> <ul style="list-style-type: none"> -de créer une instance de la classe EtalagePermanent ou EtalageSaisonnier - d'afficher toutes les caractéristiques de l'étalage dans un JOptionPane

Conception avec saisie du code :

```
import javax.swing.*;
import java.awt.*;

public class SaisieEtalage extends JPanel{
    private final static String[] typesEtalage={"Permanent", "Saisonnier"};
    private JComboBox saisieType = new JComboBox(typesEtalage);
    private JTextField saisieNuméro=new JTextField();
    private JTextField saisieSurface=new JTextField();
    private JTextField saisieInfoSupp=new JTextField();
    public SaisieEtalage()
    {
        JPanel panneauLabels = new JPanel (new GridLayout(4,1));
        panneauLabels.add(new JLabel("Type :"));
        panneauLabels.add(new JLabel("Numéro :"));
        panneauLabels.add(new JLabel("Surface :"));
        panneauLabels.add(new JLabel("Nb jours ou Zone :"));
        JPanel panneauSaisie = new JPanel (new GridLayout (4,1,5,5));
        panneauSaisie.add (saisieType);
        panneauSaisie.add (saisieNuméro);
        panneauSaisie.add (saisieSurface);
        panneauSaisie.add (saisieInfoSupp);
        setLayout(new BorderLayout(5,5));
        add (panneauLabels,BorderLayout.WEST);
        add (panneauSaisie,BorderLayout.CENTER);
    }
    public String getType()
    {
        return (String) saisieType.getSelectedItem();
    }
    public int getNuméro()
    {
        return Integer.parseInt(saisieNuméro.getText());
    }
    public float getSurface()
    {
        return Float.parseFloat(saisieSurface.getText());
    }
    public int getInfoSupp()
    {
        return Integer.parseInt(saisieInfoSupp.getText());
    }
}
```

```

import javax.swing.*;
import etal.*;
public class TestSaisieEtalage {

    public static void main(String[] args) throws ErreurSurEtalage {
        SaisieEtalage écran;
        String message;
        écran = new SaisieEtalage();
        int réponse;
        réponse = JOptionPane.showConfirmDialog(null, écran, "Saisie d'un étalage",
JOptionPane.OK_CANCEL_OPTION, JOptionPane.PLAIN_MESSAGE);
        if (réponse==JOptionPane.OK_OPTION)
        {
            message = écran.getType() + " " + écran.getNuméro() + " " + écran.getSurface() + " "
+ écran.getInfoSupp();
            JOptionPane.showMessageDialog (null,message);
        }
    }
}

```

Annexe 5 : la gestion de la persistance

Dans ce développement la gestion de la persistance ne sera pas prise en charge par une couche DAO. Les ordres SQL seront donc écrits au niveau de la couche métier dans les méthodes décrites ci-dessous.

```

public class abstract Etalage
{
    ...
    public void setNum(int leNum)
    {
        num = leNum;
    }
    public void setSurface(float laSurface)
    {
        surface = laSurface;
    }
    public abstract boolean chargerAvecId() throws ClassNotFoundException,Exception;
}
public class EtalagePermanent extends Etalage
{
    ...
    public boolean chargerAvecId() throws ClassNotFoundException, Exception
    {
        //méthode qui recherche dans la base l'étalage permanent ayant le numéro mémorisé dans la
        //variable d'instance « num » et qui valorise les autres variables d'instance si la recherche est
        //fructueuse
        ....
    }
}
public class EtalageSaisonnier extends Etalage
{
    ...
    public boolean chargerAvecId() throws ClassNotFoundException, Exception;
    {
        //méthode qui recherche dans la base l'étalage saisonnier ayant le numéro mémorisé dans la
        //variable d'instance « num » et qui valorise les autres variables d'instance si la recherche est
        //fructueuse
        ....
    }
}

```

Annexe 6 : la classe de test « TestCharger() »

```
import etal.*;
public class TestCharger
{
    public static void main (String[] args) throws ErreurSurEtalage, ClassNotFoundException, Exception{
        Etalage unEtalage ;
        unEtalage = new EtalageSaisonnier(2) ;
        if (unEtalage.chargerAvecId())
            System.out.println (unEtalage.toString() + " coût : " + unEtalage.getCoût()) ;
        else
            System.out.println ("Etalage saisonnier n°"+unEtalage.getNum()+" inexistant ") ;
        unEtalage = new EtalagePermanent(26) ;
        if (unEtalage.chargerAvecId())
            System.out.println (unEtalage.toString() + " coût :"+ unEtalage.getCoût()) ;
        else
            System.out.println ("Etalage permanent n°"+unEtalage.getNum()+"inexistant ") ;
    }
}
```

Annexe 7 : IHM d'affichage du coût d'un étalage

